4. Guillermo

Program Name: Guillermo.java **Input File: guillermo.dat**

You just started your new internship, and your boss Guillermo has asked you to schedule several 1 hour meetings with the team on different days. The problem is that this is a remote team and so your coworkers work all over the globe. And, being an intern and all, you feel as though if you can't schedule these meetings your return offer is as good as gone and you'll have to work at McDonalds. The time zones of each of your coworkers vary from
UTC −12:00 to UTC +14:00. A person's availability may span multiple time windows throughout the day and may even wrap past midnight.

Input: The first line contains an integer N ($1 \leq N \leq 50$) denoting the number of days your boss wants a meeting scheduled. Each day starts with a single integer P ($1 \leq P \leq 50$) - the number of people. Each person's data will be given as a JSON object (see example input) describing their timezone offset in the format ±HH:MM and their availability throughout the day.

Output: For each day, output the lexicographically smallest 1 hour meeting time in the format HH:MM − HH:MM in UTC time zone. If no such slot exists, output "`The fries are in the bag.`" to prepare for your new job since you won't be getting a return offer.

Sample input:

```
1
3
{
      "timezone": "+02:00",
      "availability": [("09:00", "12:00"), ("14:00", "18:00")]
},
{
      "timezone": "-04:00",
      "availability": [("08:00", "11:00"), ("13:00", "16:00")]
},
{
      "timezone": "+00:00",
      "availability": [("13:00", "17:00")]
}
```

Sample output:

```
13:00-14:00
```

Explanation:

```
Coworker1 (+02:00)
   ● 09:00-12:00 local -> 07:00-10:00 UTC
   ● 14:00-18:00 local -> 12:00-16:00 UTC
Coworker2 (-04:00)
   ● 08:00-11:00 local → 12:00-15:00 UTC
   ● 13:00-16:00 local → 17:00-20:00 UTC
Coworker3 (+00:00, already UTC)
   ● 13:00-17:00
```

The earliest 1 hour overlap of these 5 time windows is 13:00-14:00.